PRIMITIVE SENTENCE LOGIC RULES FOR NATURAL DEDUCTION

Conjunction
Introduction
\[
\begin{array}{c}
P \\
Q \\
\hline
P \land Q
\end{array}
\]

Elimination
\[
\begin{array}{c}
P \land Q \\
\hline
P
\end{array}
\] \hspace{1cm} \begin{array}{c}
P \land Q \\
\hline
Q
\end{array}

Disjunction
Introduction
\[
\begin{array}{c}
P \\
\hline
P \lor Q
\end{array}
\]

Elimination
\[
\begin{array}{c}
P \lor Q \\
\hline
P
\end{array}
\] \hspace{1cm} \begin{array}{c}
P \lor Q \\
\hline
Q
\end{array}

Conditional
Introduction
\[
\begin{array}{c}
\neg P \\
Q \\
\hline
P \rightarrow Q
\end{array}
\]

Elimination
\[
\begin{array}{c}
P \rightarrow Q \\
Q \\
\hline
P
\end{array}
\] \hspace{1cm} \begin{array}{c}
P \rightarrow Q \\
\neg Q \\
\hline
\neg P
\end{array}

Biconditional
Introduction
\[
\begin{array}{c}
P \leftrightarrow Q \\
\hline
P \leftrightarrow Q
\end{array}
\]

Elimination
\[
\begin{array}{c}
P \leftrightarrow Q \\
\hline
P
\end{array}
\] \hspace{1cm} \begin{array}{c}
P \leftrightarrow Q \\
\hline
Q
\end{array}

Negation
Introduction
\[
\begin{array}{c}
P \\
\hline
\neg P
\end{array}
\]

Elimination
\[
\begin{array}{c}
\neg P \\
\hline
P
\end{array}
\] \hspace{1cm} \begin{array}{c}
\neg P \\
Q \\
\hline
\neg Q
\end{array}

Reiteration: If a sentence occurs, either as a premise or as a conclusion in a derivation, that sentence may be copied (reiterated) in any of that derivation's LOWER sub-derivations, or lower down in the same derivation.

DERIVED SENTENCE LOGIC RULES FOR NATURAL DEDUCTION

Argument by Cases
\[
\begin{array}{c}
P \\
\neg Q \\
\hline
R
\end{array}
\]

Biconditional
Introduction
\[
\begin{array}{c}
P \\
\neg Q \\
\hline
P \leftrightarrow Q
\end{array}
\]

Biconditional
Elimination
\[
\begin{array}{c}
P \leftrightarrow Q \\
\hline
P
\end{array}
\] \hspace{1cm} \begin{array}{c}
P \leftrightarrow Q \\
\hline
Q
\end{array}

Disjunction
Elimination
\[
\begin{array}{c}
P \lor Q \\
\hline
P
\end{array}
\] \hspace{1cm} \begin{array}{c}
P \lor Q \\
\hline
Q
\end{array}

Reductio Ad Absurdum
\[
\begin{array}{c}
P \\
\neg P \\
\hline\text{Contradiction}
\end{array}
\]

Weakening
\[
\begin{array}{c}
P \\
\hline
Q
\end{array}
\]

Denying the Consequent
\[
\begin{array}{c}
P \\
\neg Q \\
\hline
\neg P
\end{array}
\]

Negation
Introduction
\[
\begin{array}{c}
\neg P \\
\hline
\neg P
\end{array}
\]

Contradiction
\[
\begin{array}{c}
\neg P \\
\hline
\neg P
\end{array}
\]

De Morgan's Rules
\[
\neg (P \lor Q) \leftrightarrow \neg P \land \neg Q 
\]

Contrapositive
\[
\neg (P \rightarrow Q) \leftrightarrow \neg Q \rightarrow \neg P 
\]

Conditional Rules
\[
\neg (P \rightarrow Q) \leftrightarrow \neg P \lor Q 
\]

Reiteration: If a sentence occurs, either as a premise or as a conclusion in a derivation, that sentence may be copied (reiterated) in any of that derivation's LOWER sub-derivations, or lower down in the same derivation.
PRIMITIVE PREDICATE LOGIC RULES FOR NATURAL DEDUCTION

Universal Introduction

\[ (\ldots \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

Universal Elimination

\[ (\forall x)(\ldots x \ldots) \]
\[ (\ldots x \ldots) \]

where \( x \) occurs arbitrarily and \( (\forall x)(\ldots x \ldots) \) is the universal generalization of \( (\ldots x \ldots) \)

Existential Introduction

\[ (\ldots x \ldots) \]
\[ (\exists x)(\ldots x \ldots) \exists \]

Where \( (\exists x)(\ldots x \ldots) \) is an existential generalization of \( (\ldots x \ldots) \)

Existential Elimination

\[ (\exists x)(\ldots x \ldots) \]
\[ s \]
\[ (\ldots x \ldots) \]
\[ x \]

Where \( (\ldots x \ldots) \) is a substitution instance of \( (\exists x)(\ldots x \ldots) \) and \( s \) is isolated in the subderivation

DERIVED RULES FOR NEGATED QUANTIFIERS

\[ \neg (\forall x)(\ldots x \ldots) \]
\[ (\exists x)(\ldots x \ldots) \]
\[ \neg (\exists x)(\ldots x \ldots) \]

\[ \neg (\forall x)(\ldots x \ldots) \]
\[ (\exists x)(\ldots x \ldots) \]

SENTENCE LOGIC TRUTH TREE RULES

Rule \&

\[ X \land Y \]
\[ \neg (X \land Y) \]

Rule \lor

\[ X \lor Y \]
\[ \neg (X \lor Y) \]

Rule \implies

\[ X \implies Y \]
\[ \neg (X \implies Y) \]

Rule \equiv

\[ X \equiv Y \]
\[ \neg (X \equiv Y) \]

Rule \neg

\[ \neg X \]

Rule \exists

\[ (\exists x)(\ldots x \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

Rule \forall

\[ (\forall x)(\ldots x \ldots) \]
\[ (\exists x)(\ldots x \ldots) \]

Rule \neg \exists

\[ (\exists x)(\ldots x \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

Rule \exists \forall

\[ (\forall x)(\ldots x \ldots) \]
\[ (\exists x)(\ldots x \ldots) \]

Rule \forall \exists

\[ (\exists x)(\ldots x \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

DERIVED RULES FOR NEGATED QUANTIFIERS

(PREDICATE LOGIC TRUTH TREE RULES)

Rule \forall

\[ (\forall x)(\ldots x \ldots) \]
\[ (\exists x)(\ldots x \ldots) \]

Rule \exists

\[ (\exists x)(\ldots x \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

Rule \neg \exists

\[ (\exists x)(\ldots x \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

Rule \exists \forall

\[ (\forall x)(\ldots x \ldots) \]
\[ (\exists x)(\ldots x \ldots) \]

Rule \forall \exists

\[ (\exists x)(\ldots x \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

DERIVED RULES FOR NEGATED QUANTIFIERS

(PREDICATE LOGIC TRUTH TREE RULES)

Rule \forall

\[ (\forall x)(\ldots x \ldots) \]
\[ (\exists x)(\ldots x \ldots) \]

Rule \exists

\[ (\exists x)(\ldots x \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

Rule \neg \exists

\[ (\exists x)(\ldots x \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

Rule \exists \forall

\[ (\forall x)(\ldots x \ldots) \]
\[ (\exists x)(\ldots x \ldots) \]

Rule \forall \exists

\[ (\exists x)(\ldots x \ldots) \]
\[ (\forall x)(\ldots x \ldots) \]

NATURAL DEDUCTION RULES FOR IDENTITY AND FUNCTION SYMBOLS

Rule \neq

\[ s \neq t \]
\[ s = t \]

Rule \equiv

\[ s = t \]
\[ s = t \]

(s a NEW NAME)

NATURAL DEDUCTION RULES FOR IDENTITY AND FUNCTION SYMBOLS

Rule \neq

\[ s \neq t \]
\[ s = t \]

Rule \equiv

\[ s = t \]
\[ s = t \]

(s a NEW NAME)

TRUTH TREE RULES FOR IDENTITY AND FUNCTION SYMBOLS

Rule \neq:

Close any branch on which \( s \neq s \) appears.

Rule \equiv:

When \( s = t \) appears on a branch, substitute \( s \) and \( t \) for each other wherever possible, without checking the resulting lines.

For function symbols: Instantiate all universally quantified sentences with all constant terms on the branch. In existentially quantified sentences use only one NEW NAME.